Particle flow for nonlinear filters with log-homotopy

Fred Daum & Jim Huang

ABSTRACT

We describe a new nonlinear filter that is vastly superior to the classic particle filter. In particular, the computational complexity of the new filter is many orders of magnitude less than the classic particle filter with optimal estimation accuracy for problems with dimension greater than 2 or 3. We consider nonlinear estimation problems with dimensions varying from 1 to 20 that are smooth and fully coupled (i.e. dense not sparse). The new filter implements Bayes' rule using particle flow rather than with a pointwise multiplication of two functions; this avoids one of the fundamental and well known problems in particle filters, namely "particle collapse" as a result of Bayes' rule. We use a log-homotopy to derive the ODE that describes particle flow. This paper was written for normal engineers, who do not have homotopy for breakfast.

KEY WORDS

particle filter, curse of dimensionality, particle collapse, degeneracy, nonlinear filter, Kalman filter, homotopy, concentration of measure, Metropolis-Hastings, MCMC

1.0 INTRODUCTION

We derive a new nonlinear filter that is vastly superior to the classic particle filter (see Figures 1 and 2). We use a log-homotopy to implement Bayes' rule with particle flow rather than with a pointwise multiplication of two functions. Homotopy allows us to migrate the particles smoothly, thereby avoiding the well known and fundamental problem with particle filters, namely "particle collapse" as a result of Bayes' rule. This problem is described in detail in [3] and [7], where it is called "degeneracy" (see pages 40-44 in [3]). Figure 3 shows the particle collapse that occurs with standard particle filters. In particular, the particles are well distributed to represent the prior probability density of the state vector, but when we get a new measurement, the likelihood of the measurement is not well represented by this set of particles (i.e., there are no particles near the peak of the likelihood function). In low dimensional

Signal and Data Processing of Small Targets 2008, edited by Oliver E. Drummond
Proc. of SPIE Vol. 6969, 696918, (2008) · 0277-786X/08/$18 · doi: 10.1117/12.764909

problems, it is easy to sample new particles from the likelihood itself to avoid this problem, but in higher dimensions, this idea does not work. Moreover, the effect of particle collapse is much more severe as the dimension of the state vector grows. Also, particle collapse is worse for more accurate measurements (i.e., at higher signal to noise ratio), as noted in [7]. This is easy to understand intuitively; imagine the likelihood function getting narrower and narrower, and as a result, the particles used to represent the prior density are getting worse and worse at representing the likelihood (i.e., there are no particles near the peak of the likelihood function). Figure 3 shows an example of particle collapse for a low dimensional example (d =2). Particle flow induced by our log-homotopy solves this problem by smoothly transforming the set of particles used to represent the prior density into a new set of particles that is good for representing the product of the prior and the likelihood (Bayes' rule), as shown in Figure 4. The arrows show the direction of flow of each particle, and the speed of flow is represented by the length of each arrow in Figure 4.

The fundamental difficulty in high dimensions is that we cannot select a good set of points to represent a function without somehow representing the function to begin with. That is, we have a chicken and egg problem, which homotopy solves by representing the two factors of the function, and connecting them with an ODE. In other words, the homotopy smoothly transforms a set of points (used to represent the prior density of the state) into another set of points that is specifically designed to represent the desired product of two known functions (the posteriori density of the state). This seems like magic, but it is actually only calculus and linear algebra. It turns out that a homotopy for the conditional probability density itself does not work, owing to a singularity in the initial condition of the ODE, whereas a log-homotopy removes this singularity and works extremely well.

We compare the computational complexity of our new algorithm with a set of carefully designed particle filters for an interesting class of smooth but dense nonlinear filter problems of increasing dimension, in which filter performance is normalized to optimal estimation accuracy, similar to the plots in [9]. This class of nonlinear problems is obtained by smoothly transforming linear filter problems using a C^2 diffeomorphism; we use this class of problems because the optimal estimation accuracy is easily computed for the linear problem using a Kalman filter, and we can compute the corresponding optimal accuracy for the nonlinear problem by a simple transformation of the covariance matrix. We are careful to maintain 20 dB signal-to-noise ratio (SNR) as dimension is varied, to avoid using problems that are either too easy or too hard. It turns out that testing nonlinear filters for certain applications with

measurements at very low SNR makes the curse of dimensionality seem less fearsome, owing to the broader measurement likelihoods; in fact, this is a standard trick in particle filter papers to make some algorithms look good. Our plots are normalized to optimal filter performance, as defined in [9], in contrast to most papers on particle filters, which merely show accuracy unrelated to optimal performance or else convergence (to something) without reference to optimal accuracy. For a certain class of smooth but dense nonlinear & non-Gaussian filtering problems, that enjoys concentration of measure (similar to Gaussian densities), our new algorithm is vastly superior to the classic particle filter, without requiring the use of a good proposal density supplied by an EKF or UKF as an input to the algorithm (e.g., see [4]). If the EKF or UKF can supply a good proposal density, then the particle filter in [4] gives excellent results, as shown in [9]; however, when a good proposal density is not available (e.g., because the EKF and UKF diverge or are highly suboptimal), then particle filters such as in [4] suffer from the curse of dimensionality (see [9]).

Figure 1 summarizes our results for the new log-homotopy filter, which are vastly superior to the classic particle filter shown in Figure 2. We have plotted the median chi-square errors in the d-dimensional state vector over 100 Monte Carlo runs, and we have normalized our results with respect to optimal estimation accuracy as well as dimension (d), so that unity in these plots corresponds to optimal filtering for any dimension (see [9]). Figure 2 clearly shows that the classic particle filter suffers from the curse of dimensionality, as explained theoretically in [9].

A survey of the nonlinear filter problem along with various solutions is given in [10], and our notation and technical assumptions are all in [10]. An excellent and extremely lucid textbook on particle filters with many detailed examples is [3]. The seminal paper on particle filters is [1], and many other useful papers on particle filters are in [2] & [5]. It has been asserted that particle filters beat the curse of dimensionality [6], but this is generally wrong, as shown in [9].

2.0 Derivation of log-homotopy

We start by recalling that the unnormalized conditional probability density of the d-dimensional state vector x is computed using Bayes' rule as follows:

$$p(x, t_k | Z_k) = p(z_k | x, t_k) p(x, t_k | Z_{k-1}) \qquad (1)$$

in which

$z_k = k^{th}$ measurement

$Z_k = \{z_1, z_2, ..., z_k\}$

$p(z_k | x, t_k) =$ probability density of measurement z_k at time t_k conditioned on x

In particle filters, as well as essentially any other nonlinear filter, Bayes' rule is implemented as a pointwise multiplication of two functions; this is the obvious way to multiply two functions, and it is the way that any normal person would do it. However, there is a very serious and ubiquitous problem with this straightforward approach; namely, that the representation of this product suffers from the curse of dimensionality. In the literature of particle filters, the effect of this problem is called "particle collapse". In contrast, the solution of the **F**okker-Planck equation does not suffer from this problem, owing to the use of particle flow, wherein the particles migrate smoothly in time according to the system dynamics, and the resulting distribution of particles in state space does not suffer from particle collapse. Our fundamental idea is to create a differential equation to implement Bayes' rule, and induce a flow of particles analogous to the natural flow in time induced by the Fokker-Planck equation. That is, we want to find an ODE to implement Bayes' rule, rather than the pointwise multiplication of two functions. We will use a homotopy to create this ODE. We cannot create a flow in time, because Bayes' rule operates at discrete points in time; however, we can introduce a scalar valued parameter, called λ, that plays the role of time, and which varies from 0 to 1. You can think of λ as a little loop of synthetic time, which we insert at each discrete measurement time. You should not be afraid of the word "homotopy" as used here; we do not need any abstract mathematics, and you do not need to know the finer points of topology, but rather we will use an elementary definition and simple high school algebra that any normal engineer can readily grasp. Homotopy methods are routinely used to solve systems of multivariate polynomial equations and other practical problems (see [8] and [11] for surveys which are readable by normal engineers).

We will use shorthand notation for Bayes' rule, because we are only interested in the fact that we are trying to compute the product of two functions:

$$f(x) = g(x)h(x) \tag{2}$$

in which g(x) is the prior density of the d-dimensional state vector x, and h(x) is the likelihood of the measurement. We will suppress the argument (x) in the following:

$$f = gh \tag{3}$$

First, take the logarithm of both sides:

$$\log(f) = \log(g) + \log(h) \tag{4}$$

Because the functions of interest are probability densities or likelihoods, these are positive functions on the appropriate domain (i.e., the intersection of the supports of g and h), and therefore the logarithms are finite and real valued on the relevant domain.

Second, we will define a homotopy function as follows:

$$\log(f_\lambda) = \log(g) + \lambda \log(h) \tag{5}$$

in which λ is a real valued parameter that varies between 0 and 1. For λ = 1, the homotopy function is exactly equal to log(f), which is what we want to compute, whereas for λ = 0, the homotopy function is exactly equal to log(g).

Third, we can create an ODE by differentiating with respect to λ:

$$\frac{\partial \log(f_\lambda)}{\partial \lambda} = \log(h) \tag{6}$$

Fourth, we can explicitly and exactly compute the flow of the particles corresponding to the above homotopy using the chain rule and the generalized inverse as follows. Using the chain rule on our log-homotopy function, we obtain:

$$\frac{d \log(f_\lambda)}{d\lambda} = \frac{\partial \log(f_\lambda)}{\partial x} \frac{dx}{d\lambda} + \frac{\partial \log(f_\lambda)}{\partial \lambda} \tag{7}$$

Following Liouville's approach to deriving an ODE for particle flow in physics, we are trying to move the particles so that the relevant function values stays constant.

Therefore, the flow of each particle is characterized by the condition that the total derivative of the relevant function with respect to λ is zero:

$$\frac{d\log(f_\lambda)}{d\lambda} = \frac{\partial \log(f_\lambda)}{\partial x}\frac{dx}{d\lambda} + \frac{\partial \log(f_\lambda)}{\partial \lambda} = 0 \qquad (8)$$

For d = 1 we can compute the flow of particles by solving (8) exactly as follows:

$$\frac{dx}{d\lambda} = -\frac{\partial \log(f_\lambda)}{\partial \lambda} \Big/ \frac{\partial \log(f_\lambda)}{\partial x} \qquad (9)$$

for non-zero gradient and dx/dλ = 0 otherwise. However, for d > 1, we cannot simply solve equation (8) by division as above, but rather we can find the unique minimum norm solution using the generalized inverse as follows:

$$\frac{dx}{d\lambda} = -\left(\frac{\partial \log(f_\lambda)}{\partial x}\right)^{\#} \frac{\partial \log(f_\lambda)}{\partial \lambda} \qquad (10)$$

in which A# denotes the generalized inverse of A, which can be explicitly computed in this case as follows [14]:

$$A^{\#} = A^*(AA^*)^{-1} = A^T / AA^T \qquad (11)$$

assuming that AA* > 0, and A# = 0 otherwise, where A* denotes the adjoint (or transpose) of the matrix A. Finally, the induced particle flow can be simplified to:

$$\frac{dx}{d\lambda} = -\log(h)\left(\frac{\partial \log(f_\lambda)}{\partial x}\right)^T \Big/ \left\|\frac{\partial \log(f_\lambda)}{\partial x}\right\|^2 \qquad (12)$$

for non-zero gradient and dx/dλ = 0 otherwise.

There is a simple physical interpretation of the flow of particles that we have just derived. In particular, the induced flow of particles is in the gradient direction of the log-homotopy, with speed proportional to log(h), and with the important condition that the flow stops when the gradient is zero. We can numerically integrate these

ODEs, (6) and (12), jointly from $\lambda = 0$ to $\lambda = 1$, for each point x in state space, with the initial condition on (6) of log(g) at $\lambda = 0$, and the result is that we have computed log(f) as desired. The crucial property of these ODEs is that we migrate the particles (i.e., points in state space) as we integrate (12) from $\lambda = 0$ to $\lambda = 1$, which allows us to maintain a good set of particles to represent the product of g & h, analogous to the flow of particles that is a good representation of the solution of the Fokker-Planck equation. The end result of this process is a good set of particles to represent the product of g & h, and therefore we can simply use this good set of particles to perform a pointwise multiplication of g & h, thereby avoiding the loss of accuracy as a result of the numerical integration of (6) itself.

The condition for $A\# = 0$ stated after (11) is not merely an academic mathematical detail, but rather it is important for numerical computations; we discovered this the hard way. It turns out that we need to enforce a lower bound on AA^* that is many orders of magnitude bigger than the LSB of our computer; that is, the actual computation is: $A\# = A^*/(AA^*)$ for $AA^* > c$, and $A\# = 0$ otherwise. We need a surprisingly large value of c even for d = 1 problems; otherwise we get a few percent outliers that ruin the results. This is similar to the acceptance threshold in Monte Carlo methods such as Metropolis-Hastings [13], except that our algorithm is not random, in contrast with Monte Carlo methods. It is extremely interesting that our new algorithm appears to be something like a deterministic version of Metropolis-Hastings.

In other practical applications the generalized inverse is notoriously bad, resulting in a formal theoretical solution which turns out to be useless for numerical algorithms; however, here we get a physically meaningful and numerically robust formula (after the appropriate engineering to select the value of c). It is very interesting that we obtain a gradient flow in the log-homotopy, similar to diffusion sampling Monte Carlo methods (e.g, see [13]), which uses a gradient flow of the log-density. We emphasize that our log-homotopy is not a Monte Carlo method, but rather it is completely deterministic and does not use any random numbers. Moreover, we have no free design parameters that need to be adjusted or adapted, in strong contrast to [13]. On the other hand, in addition to the unique minimum norm solution derived above, there is an infinite number of other solutions of equation (7) that can be explicitly computed as follows (see page 22 of [14]):

$$\frac{dx}{d\lambda} = -A^{\#} \frac{\partial \log(f_\lambda)}{\partial \lambda} + [I - A^T A / Tr(A^T A)]y \qquad (13)$$

in which y is an arbitrary d-dimensional vector, and

$$A = \frac{\partial \log(f_\lambda)}{\partial x} \qquad (14)$$

Such additional solutions might be exploited to enhance computational efficiency by optimizing the gratuitous vector y. For example, one could pick y as a Gaussian zero mean random variable, with optimal covariance matrix, analogous to the optimal scaling for Metropolis-Hastings in [13]. Don't be too surprised if you see a paper that does exactly this in the near future.

The beautiful paper by Mathe´ and Novak [17] shows that Monte Carlo algorithms generally suffer from the curse of dimensionality, and to have any chance of beating this curse one needs to use adaptive algorithms for problems with "good" structure. The specific "good" structure assumed in [17] is: a log-concave probability density on the unit ball in d-dimensional Euclidean space with bounded Lipschitz constant (α) of the log-density; this class obviously includes all Gaussian densities as well as other densities that enjoy concentration of measure, such as those that we called "vaguely Gaussian" in [9]. In particular, Theorem 5 in [17] gives an amazingly explicit upper bound on the variance of estimation error, which is at worst quadratic in d and α. It is easy to show that the natural exponential family is log-concave, but we know that we can design exact finite dimensional nonlinear filters for the exponential family, and thus beat the curse of dimensionality [10]; hence, Theorem 5 of Mathe´ and Novak is perhaps not too surprising. However, this theoretical bound is asymptotic in the number of Monte Carlo samples, so its relevance to practical algorithms is conjectural. Moreover, the design of robust adaptive algorithms for these problems is easier said than done [18]-[19].

REFERENCES:

[1] N. J. Gordon, D. J. Salmond and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," IEE Proceedings-F 140(2), pp. 107-113 (1993).

[2] "Sequential Monte Carlo Methods in Practice," edited by A. Doucet, N. de Freitas and N. Gordon, Springer-Verlag, 2001.

[3] B. Ristic, S. Arulampalam and N. Gordon, "Beyond the Kalman filter," Artech House, 2004.

[4] R. van der Merwe, A. Doucet, N. de Freitas and E. Wan, "The Unscented Particle Filter," Cambridge University Technical Report (TR 380), August 2000.

[5] "Special Issue on Monte Carlo Methods for Statistical Signal Processing, " IEEE Transactions on Signal Processing, February 2002.

[6] D. Crisan and A. Doucet, "A Survey of Convergence Results on Particle Filtering Methods for Practitioners," IEEE Transactions on Signal Processing, March 2002.

[7] Mike Pitt and Neil Shepard, "Filtering via simulation: auxiliary particle filters," Journal of American Statistical Association, pages 590-599, June 1999.

[8] Hansjörg Wacker, "A summary of the developments in imbedding methods," Chapter 1 in "Continuation Methods," edited by H. Wacker, Academic Press, 1978.

[9] F. E. Daum and J. Huang, "Curse of Dimensionality and Particle Filters," Proceedings of IEEE Aerospace Conference, Big Sky Montana, March 2003.

[10] F. E. Daum, "Nonlinear filters: Beyond the Kalman filter," Special tutorial issue of the IEEE Aerospace and Electronic Systems Magazine, August 2005.

[11] T. Y. Li, "Numerical solution of multivariate polynomial systems by homotopy continuation methods," Acta Numerica Volume 6, Cambridge University Press, 1997.

[12] S. Smale, "Complexity theory and numerical analysis," op. cit.

[13] G. O. Roberts and J. S. Rosenthal, "Optimal Scaling for Various Metropolis-Hastings Algorithms," Journal of Statistical Science, Volume 16 Number 4, 2001.

[14] C. R. Rao and S. K. Mitra, "Generalized inverse of matrices and its application," John Wiley & Sons, Inc., 1971.

[15] Ernest Weekley, "Etymological Dictionary of Modern English," page 230, Dover Books, 1967.

[16] Tom Kailath, "The innovations approach to detection and estimation theory," page 681, Proceedings of IEEE, Volume 58 Number 5, May 1970.

[17] Peter Mathe´ and Erich Novak, "Simple Monte Carlo and the Metropolis Algorithm," submitted to Journal of Complexity, 1 June 2007.

[18] Gareth Roberts and Jeff Rosenthal, "Coupling and convergence for MCMC," Workshop on new directions in Monte Carlo methods, Fleurance France, 25 June 2007.

[19] Mylene Bedard and Jeff Rosenthal, "Optimal scaling of Metropolis algorithms: is 0.234 as robust as is believed?" preprint, July 2007.

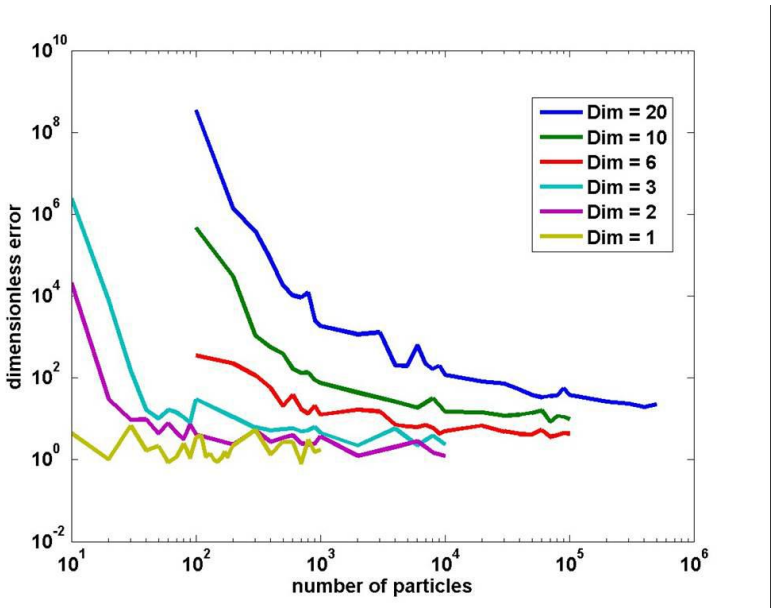

Figure 1 Log-homotopy particle filter

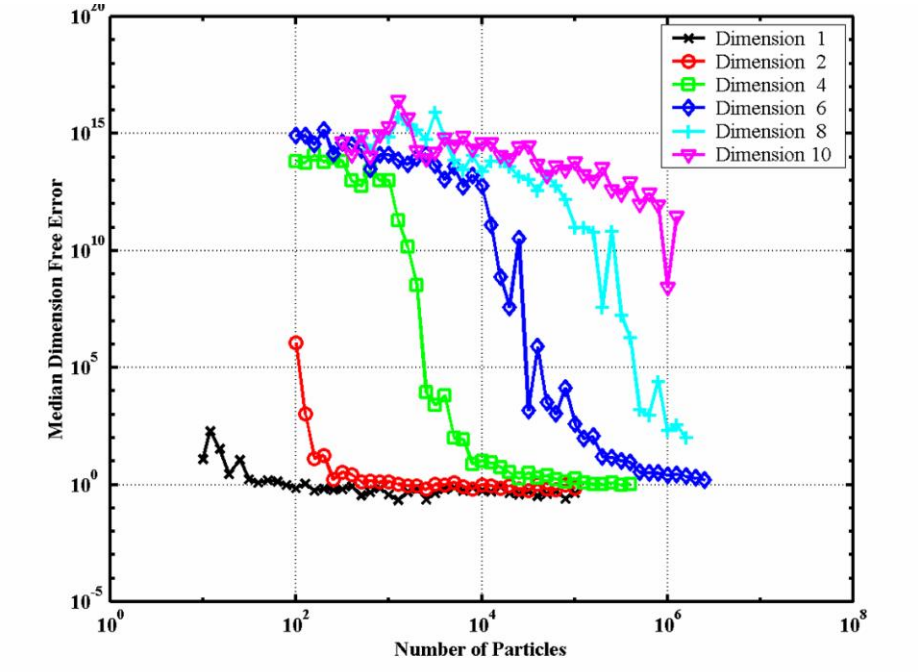

Figure 2 Curse of dimensionality for classic PF

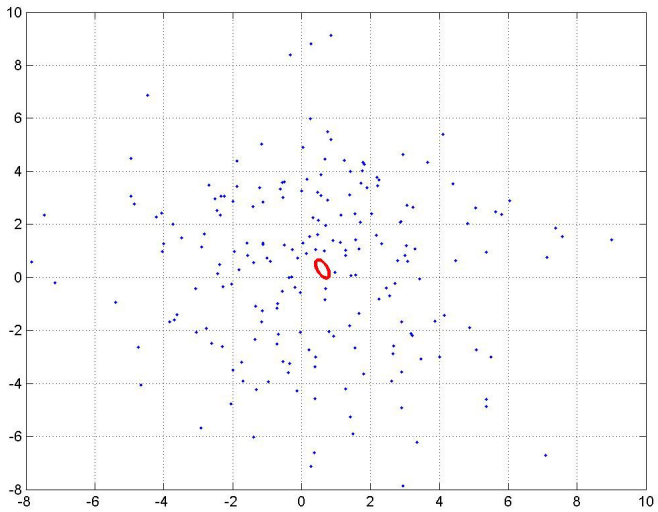

Figure 3 Particle Collapse or "degeneracy": the red ellipse is the one-sigma contour for the likelihood and the dots are the particles for the prior density; these particles cannot be used for a good approximation of the posteriori density (i.e. the product of the prior and the likelihood).

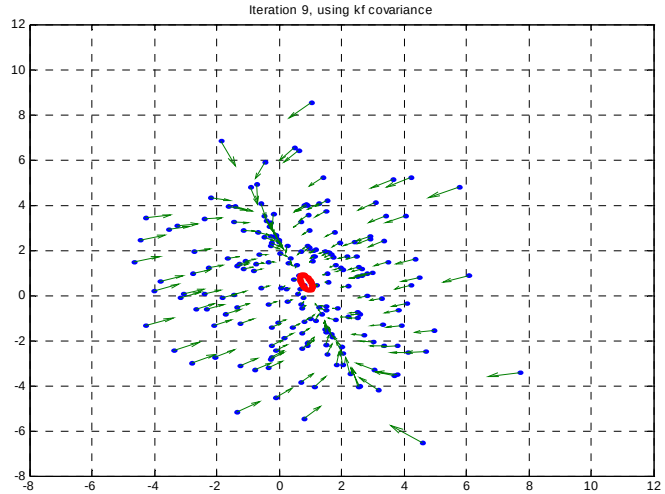

Figure 4 Particle flow induced by log-homotopy: the arrows show the direction and speed of flow of each particle